User-Centered Design: Component-Based Web Technology

Esin Kiris, Howard Abrams, and Roman Longoria

CA One CA Plaza
Islandia, NY 11749, USA
esin.kiris@ca.com, howard.abrams@ca.com, roman.longoria@ca.com

Abstract. In this age of rapid technological progression and heightened competition, designers of interactive systems, especially web applications, must be able to prepare for, cope with, and adopt to design processes that meet both customer needs and expectations and cutting edge-technology. This paper presents the authors' experience with designing and prototyping a web application using a new web user interface (UI) development technology. We describe how the technological progression forced significant changes in User-Centered Design (UCD) process and design tools. We then discuss the contributions of these changes to the design and development of an internationalized web application. We provide background information about an *Abstract UI* and the web implementation using JavaServer Faces (JSF)[1] technology. We describe how this new technology will be adopted into CA's UCD process and present a case study in which the new JSF technology solution is used for a prototype of an enterprise storage management application. We then discuss the pros and cons of using this technology at the design stage, providing some structure and guidance to designers who might be faced with similar situations. This paper suggests there may be a more appropriate alternative to the current design processes and tools used for designing web applications.

Keywords: UI technology, Java Server Faces (JSF), Prototyping, User-Centered Design, Internationalization, AJAX.

1 Introduction

In technologically-developed countries, people have become increasingly reliant on electronically-delivered information and services as information technology is embedded into more and more every day items. Users' expectations of the interactive performance of these items have also increased. Today's users expect to interact with electronic applications like they interact with the desktop applications – simply and intuitively.

To meet user expectations, web UI technology has been rapidly changing and web applications are providing more and more interactive UI functions and features similar to desktop applications. In other words, web application UIs are becoming "rich" interfaces. A rich interface reduces server round-trips, receives data from the server without a full page refresh, and increases interactivity with desktop-like interaction. The new goal of web applications is to be as rich as possible. The web technologies used to create rich and universal web applications presented in this paper are JavaServer Faces (JSF) [1] and Asynchronous JavaScript and XML (AJAX).

This paper evaluates the JSF technology and new prototyping tool, Exadel Studio Pro[1] ™. In addition, a case study is presented on a pilot project which examined the implications for the web application design process and UCD methods.

2 Background

Several years ago, CA undertook the ambitious goal of creating a single UI look and feel across hundreds of products. One result of this effort is a large and continuously updated set of UI standards for the products.

Early in the standardization process it was realized that, without a set of reusable technologies, it would be extremely costly to implement any standard, let alone one as detailed as the company's. The technology standardization process started with typically web application artifacts: CSS stylesheets and Java Server Page (JSP) tags. But the limits of these relatively simple technologies became evident in the face of the evolving, complex UI standards. At the same time, there were non-web-based products that also needed a consistent look and feel.

To overcome these obstacles, CA has developed an Abstract UI. An Abstract UI is a defined, consistent set of declarative application programming interfaces (APIs), independent of the underlying technology and the UI's look and feel. This technique allows the technology and look of the UI to change with limited impact on the development team. It also allows UI designers to develop UIs that can quickly and easily be reused. The downside to this approach is the need to develop and maintain the API and a set of implementing technologies

3 UI Technology

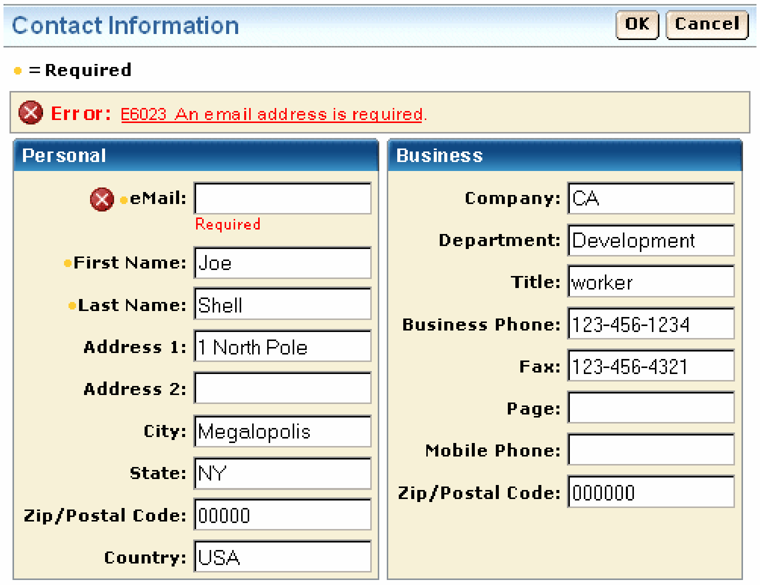
The current Abstract UI implementation for web-based applications uses JSF as its underlying technology. While JSF is a relatively new J2EE standard for web-based UIs, it was chosen for three key concepts that it employs: components, renderers, and tool support. Instead of building a user interface with raw HTML, CSS, and JavaScript, JSF uses reusable components, like traditional desktop-based applications. Rendering is separated from the model of the component, allowing the rendering to change over time without affecting the developer's use of the component when programming. This separation can also mean the same UI could be rendered in two different styles without modifying the application. In Addition, this separation enables UI to be universal where UI can be localized into several languages [2]. Tool support was also an important consideration during the design of the JSF framework. Since tooling was standard, there were many tool venders to choose from. WYSIWYG design tools were important since they let UI designers, who are not trained as

[1] Trademark or registered trademark of Exadel, Inc.

programmers, build working UIs that development teams can then hook up to live data. This means the actual UI can be tested by users during UI design stage without having to build the application. This is a key feature in CA's overall UCD program.

To implement the Abstract UI and CA UI Standards, a comprehensive set of reusable JSF UI components were created. The rationale behind creating a set of CA-specific components rather than re-style a set of generic third party components was that CA's UI Standards are very specific and detailed, allowing the custom components to encapsulate knowledge of the standards beyond simple colors and fonts. The resulting reusable components encapsulate complex rules, for example data validation error message and icon display. By simply placing a label and a text field in the interface, the complex rules describing how, where, and when messages and icons should be used are automatically implemented for the developer or designer. Figure 1 shows an example from CA's UI Standards on message display.

Another example is that the CA UI Standards specify that the UI designer can place anchor links at the top of the page that allow the user to jump to specific sections within the page (see Figure 2). Under each section, a 'back to top' link is placed to return the user back to the top of the page. Rather than making the developer understand the UI Standard and place each anchor and 'back to top' link, the components allow the developer to specify an anchor bar and list each section it should contain. When the page renders, it not only renders the bar in the proper colors and fonts, but can automatically place the 'back to top' links under each section, set proper tooltips, etc.

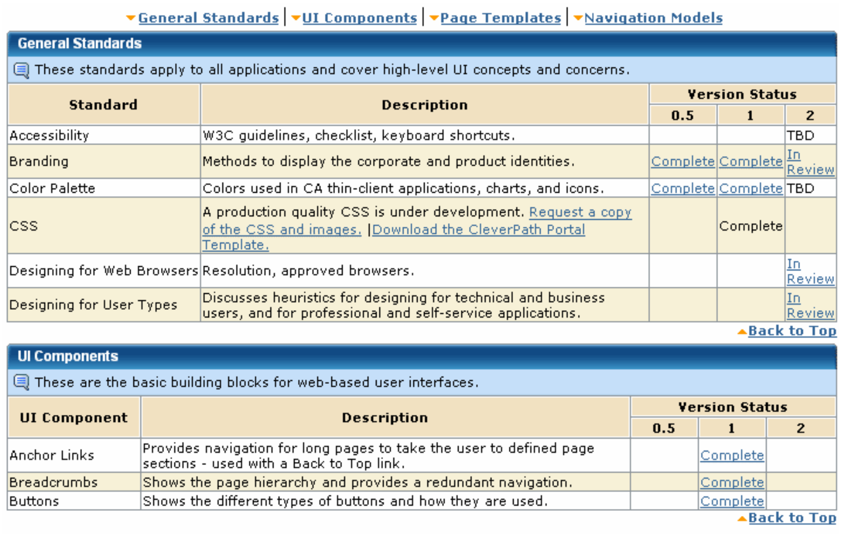


● = Required

Error: E6023 An email address is required.

Personal		Business	
●eMail:	Required	Company:	CA
●First Name:	Joe	Department:	Development
●Last Name:	Shell	Title:	worker
Address 1:	1 North Pole	Business Phone:	123-456-1234
Address 2:		Fax:	123-456-4321
City:	Megalopolis	Page:	
State:	NY	Mobile Phone:	
Zip/Postal Code:	00000	Zip/Postal Code:	000000
Country:	USA		

Fig. 1. An example of reusable components automatically implementing validation error message and icon display



General Standards | UI Components | Page Templates | Navigation Models

General Standards

These standards apply to all applications and cover high-level UI concepts and concerns.

Standard	Description	Version Status		
		0.5	1	2
Accessibility	W3C guidelines, checklist, keyboard shortcuts.			TBD
Branding	Methods to display the corporate and product identities.	Complete	Complete	In Review
Color Palette	Colors used in CA thin-client applications, charts, and icons.	Complete	Complete	TBD
CSS	A production quality CSS is under development. Request a copy of the CSS and images. |Download the CleverPath Portal Template.		Complete	
Designing for Web Browsers	Resolution, approved browsers.			In Review
Designing for User Types	Discusses heuristics for designing for technical and business users, and for professional and self-service applications.			In Review

Back to Top

UI Components

These are the basic building blocks for web-based user interfaces.

UI Component	Description	Version Status		
		0.5	1	2
Anchor Links	Provides navigation for long pages to take the user to defined page sections - used with a Back to Top link.		Complete	
Breadcrumbs	Shows the page hierarchy and provides a redundant navigation.		Complete	
Buttons	Shows the different types of buttons and how they are used.		Complete	

Back to Top

Fig. 2. An example of 'back to top' links being automatically place in the page based on the anchors links at the top

4 User-Centered Design Process and UI Technology

The UCD team saw several benefits of using CA web components during product design cycle, especially at prototyping and testing stages. The following benefits were identified:

- Prototypes would be as interactive as real applications;
- Prototype building time would be shortened;
- Prototypes would be migrated into real UI application development;
- UI localization would be done and tested during the design cycle;
- The components would meet usability and accessibility criteria.

While these benefits were exciting, there were concerns about how a UCD professional (UI designer) would be able to use these JSF web components with limited or no Java programming experience. The team hypothesized that if there was a tool that allowed users to build web pages using JSF web components without Java programming experience, then the UI designers would be able to use the web components along with JSF technology to create prototypes during the product design cycle. An evaluation study was performed to determine the JSF UI development tool suitable for the UCD organization.

4.1 Tool Selection

A UI designer and GUI developer conducted a tool evaluation study. The goal was to identify a tool that could easily be used by UI designers without Java programming experience. Since it is a new technology, tool availability was limited. Two candidate tools were identified and evaluated for ease of use from a designer's perspective. At the end of the evaluation, Exadel™ was chosen since it creates web project deployment automatically and had several features that eased UI prototyping such as a drag and drop feature for UI components, visual view mode, source code mode, outline features, and a graphical navigational model.

4.2 User-Centered Design Infrastructure Project

After Exadel™ was chosen, the UCD team planned an infrastructure project. The goal of this project was to create an infrastructure with a number of reusable template pages and applications for UI designers and change the UCD process from creating HTML prototypes to creating JSF prototypes. The initial step of this project was to engage a pilot design project and validate the selected tool and technology. An enterprise storage management web application project was selected as the pilot JSF project.

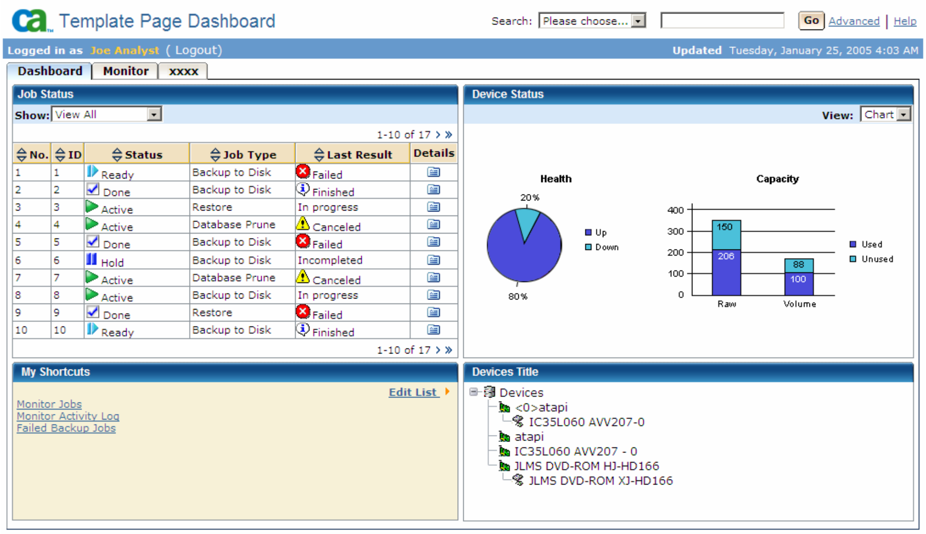
Pilot Project: An Enterprise Storage Management Web Application UI Design. The next release of enterprise storage management web application underwent a typical UCD process. This involved performing customer interviews to gather user interface requirements and determining user roles and features. After the process flow and architectural diagrams were complete, storyboards were created. A prototype of the user interface was then created using JSF technology and Exadel™. A remote usability evaluation was conducted to gather feedback on the prototype. A UI designer was assigned to this project.

JSF Prototype. Initially, the UI designer took an online training course to learn JSF technology and Exadel™. This was a week long course with a lot of hands-on lab exercises. At the end of the class, she became familiar with the tool, understood JSF technology, and was able to build a few simple JSF web applications. She spent few days on training herself on CA web components using the available documents and input from the component development team. At the end of two weeks, she became comfortable with JSF, Exadel™, and CA web components. However, she could not create Java source files for binding UI elements to the events, data, and navigational model. As a result, a Java developer joined the project to provide Java programming support. The UI designer built the presentation layer based on CA web components using Exadel™ and delivered them to the Java developer who added necessary Java source code behind the presentation layer. The result was an interactive UI prototype which could eventually be reused in application development. In addition, since the UI model was an Abstract UI where all locale-sensitive objects are separated from the core source [2], the user interface was also localized. This added value to the UCD process by potentially allowing the team to test a localized interface with users.

A remote usability evaluation was conducted to gather feedback on the JSF prototype. Since the JSF prototype was as interactive as the targeted application, the tasks that were designed to get user feedback were highly comprehensive. Some tasks included filtering, grouping, and deleting data. Participants were able to accomplish these tasks exactly as they could in the targeted application. This allowed the team to collect detailed and complete usability data on the proposed UI design.

Localization of JSF Prototype. Since localization of prototype results in cost and time savings in the development cycle, it is not common to localize prototypes at design cycle, due to the time consuming nature of this process. Most commonly, localization is left as a separate exercise at the end of development cycle during which problems resulting from word length or orientation often causes costly UI redesigns. Using JSF prototyping, it was easy to localize with no extra cost or time. This allowed the team to conduct expert design reviews and identify and solve potential localization design problems at early design stage. Although the team had a goal to conduct international usability tests, this was not accomplish due to difficulties in finding international participants as well as project time and cost limitations. However, the team did identify the most important localization issues on the user interface, such as tab and button labels and solved them before the development started.

JSF Template Pages. The pilot project results suggest that JSF Prototype has several potential benefits. However, the effort was not as minimal as it was thought at the beginning. To further streamline the process, it was decided to create reusable JSF template pages. The template pages were identified based on CA page layout standards. The identified template pages were: Dashboard (home page), Full Page with Tabs, Full Page with Tabs and Subtabs, Object Detail Page, Report Page, Wizard Drill-Down Page, and Wizard Multi-Tab Page. The UI designer reused the pilot JSF prototype and created the JSF template pages. Since no Java developer resource was available for this activity, these pages were not fully generalized in terms of Java source code. Currently, this part of the project is still in progress. However, the user interface design of template pages was completed as was the localization of the pages (see Figures 3 and 4).


Template Page Dashboard

Search: Please choose... Go Advanced | Help

Logged in as Joe Analyst (Logout) Updated Tuesday, January 25, 2005 4:03 AM

Dashboard | Monitor | xxxx

Job Status

Show: View All

1-10 of 17 > »

No.	ID	Status	Job Type	Last Result	Details
1	1	Ready	Backup to Disk	Failed	
2	2	Done	Backup to Disk	Finished	
3	3	Active	Restore	In progress	
4	4	Active	Database Prune	Canceled	
5	5	Done	Backup to Disk	Failed	
6	6	Hold	Backup to Disk	Incompleted	
7	7	Active	Database Prune	Canceled	
8	8	Active	Backup to Disk	In progress	
9	9	Done	Restore	Failed	
10	10	Ready	Backup to Disk	Finished	

1-10 of 17 > »

Device Status

View: Chart

My Shortcuts

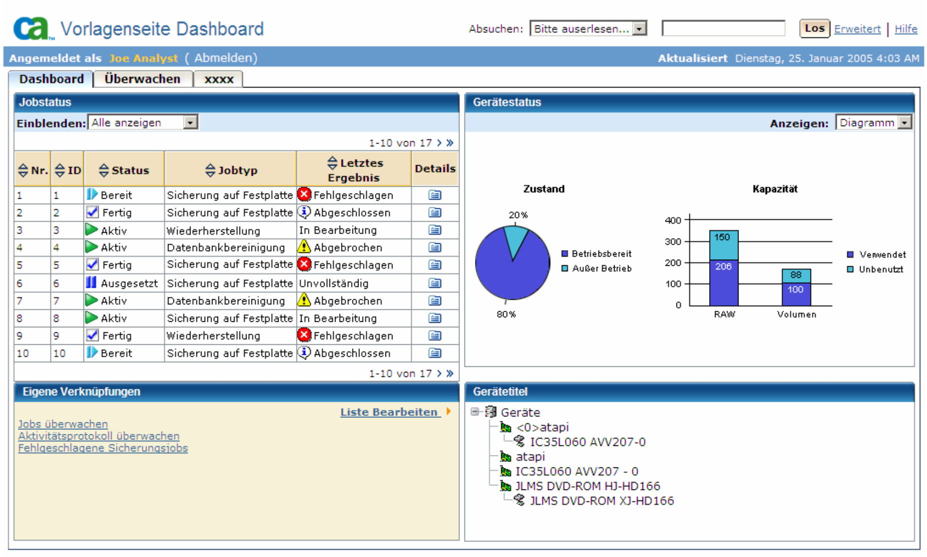
Edit List

Monitor Jobs
Monitor Activity Log
Failed Backup Jobs

Devices Title

Devices
<0>atapi
IC35L060 AVV207-0
atapi
IC35L060 AVV207 - 0
JLMS DVD-ROM HJ-HD166
JLMS DVD-ROM XJ-HD166

Fig. 3. An example JSF template page: Dashboard



Vorlagenseite Dashboard

Absuchen: Bitte auserlesen... Los Erweitert | Hilfe

Angemeldet als Joe Analyst (Abmelden) Aktualisiert Dienstag, 25. Januar 2005 4:03 AM

Dashboard | Überwachen | xxxx

Jobstatus

Einblenden: Alle anzeigen

1-10 von 17 › »

Nr.	ID	Status	Jobtyp	Letztes Ergebnis	Details
1	1	Bereit	Sicherung auf Festplatte	Fehlgeschlagen	▤
2	2	Fertig	Sicherung auf Festplatte	Abgeschlossen	▤
3	3	Aktiv	Wiederherstellung	In Bearbeitung	▤
4	4	Aktiv	Datenbankbereinigung	Abgebrochen	▤
5	5	Fertig	Sicherung auf Festplatte	Fehlgeschlagen	▤
6	6	Ausgesetzt	Sicherung auf Festplatte	Unvollständig	▤
7	7	Aktiv	Datenbankbereinigung	Abgebrochen	▤
8	8	Aktiv	Sicherung auf Festplatte	In Bearbeitung	▤
9	9	Fertig	Wiederherstellung	Fehlgeschlagen	▤
10	10	Bereit	Sicherung auf Festplatte	Abgeschlossen	▤

1-10 von 17 › »

Gerätestatus

Anzeigen: Diagramm

Eigene Verknüpfungen

Liste Bearbeiten

Jobs überwachen
Aktivitätsprotokoll überwachen
Fehlgeschlagene Sicherungsjobs

Gerätetitel

```
Geräte
  <0>atapi
    IC35L060 AVV207-0
  atapi
  IC35L060 AVV207 - 0
  JLMS DVD-ROM HJ-HD166
    JLMS DVD-ROM XJ-HD166
```

Fig. 4. An example JSF template page localized in German: Dashboard

5 Advantages and Disadvantages of JSF Prototyping

The pilot project study demonstrates that JSF prototyping has a number of advantages:

- The resulting prototype is as interactive as the targeted application.
- The prototype is the real front layer of the targeted application and is reusable in the development of application.
- Reusable CA web components are complete in terms of data validation, error messages, etc. The resulting prototype also becomes complete using these components.
- New technological updates on CA web components can automatically go into the prototypes built with them by updating the linked CA web component package.
- The UI designer directly contributes to the application development process. This eliminates the need for a detailed UI design quality test at the end of development cycle.
- The prototyping effort can be generalized using template pages. This results in savings in both cost and time in the product design and development cycles.
- The UI standards are converted into working JSF web components. This ensures the UI standards are implemented correctly and eliminates discrepancies that might be caused by different interpretations of the standards.

There are numerous challenges associated with JSF prototyping:

- Java programming experience is required to create fully interactive prototypes. This is not a typical experience owned by UI Designers.

- XML coding and simple Java data binding knowledge is needed for the UI designer.
- There is dependency on custom web components development. For any required change, the web components need to be updated first.

6 Preparing for Change

Based on the lessons learned from the pilot project, the requirements were identified for moving forward:

- Development of a JSF training program for UI designers. This program shall include the following trainings:
  - JSF UI technology training
  - CA web components training and/or tutorials for UI designers
  - Exadel tool training
- Development of CA web component working project samples.
- Availability of both a UI designer and a Java developer for any JSF prototype development. The UI designer will create the presentation layer and the Java developer will bind the presentation layer to the XML data file and navigational model.
- UI designers must understand the concept of XML and data binding.
- Simple, XML data models that allow the designer to add data to the prototype without a developer

While this change presents a challenge for UI designers, it brings significant potential benefits to the design and development process.

7 Future Direction

There is no a complete JSF prototyping tool for UI designers in the market. There is still plenty of work to be done in developing a tool that is easy to be used with no Java programming experience. As a next step, we will continue searching a complete tool suite to the UI designers' needs.

Template pages project is still in progress. The next step is to complete development of template pages and continuously enhance them in order to ease the task of creating JSF prototypes.

Another area that will continuously be renewed and maintained is CA web components. More AJAX technology will be applied to the components and they will be as "rich" as possible. In addition, bi-direction localization features will also be applied to the components

Acknowledgments. UI Web Component Development team, Rebecca Baker, and Kerry Harrison from UCD team at CA, Inc.

References

1. JavaServer Faces Technology, News and Information. Available as, http://java.sun.com/javaee/javaserverfaces
2. O'Conner, J.: Java Internationalization: Localization with ResourceBundles (1998), Available as http://java.sun.com/developer/technicalArticles/Intl/ResourceBundles